Penerapan Kombinasi Sistem Aljabar Gondran dalam Algoritma Prim pada Problem *Minimum Spanning Tree* untuk Solusi Alternatif yang Tidak Tunggal

Patricia Ardanari
Program Studi Teknik Informatika, Universitas Atma Jaya Yogyakarta
Jl. Babarsari No. 43, Yogyakarta 55281, Indonesia
E-mail: arda@staff.uajy.ac.id

Abstract. Combined Application of Gondran Algebra System In Prim's Algorithm on Minimum Spanning Tree Problem for Non Single Alternative Solution. *Nowadays technology on industrial was many that connected to the other technology. Support from technology can be seen within the method and make the procces more simple and optimal from the production section until the delivery. One example of the method are Minimum Spanning Tree (MST). This method make a better and shorter connection in all route that can make efficient for connection. In this paper, will be developed an algorithm than can give us information about problem that we face for a non single solution but also for making every MST can get their own solution. Algorithym that being used are godran aljabar and prim algorithym. To get alternative solution for each solution that came from MST problem that not singular, we combine and developed the algorithym for meet the condition to get each solution.*

Keywords: *Minimum Spanning Tree, Algoritma Prim, Aljabar Gondran, non single solution.*

Abstrak. Dewasa ini perkembangan teknologi bidang perindustrian banyak sekali yang sangat berkaitan ataupun memerlukan pendukung dari teknologi bidang lain. Pendukung teknologi dari bidang teknologi lain tersebut di antaranya adalah cara atau metode yang dapat memperlancar ataupun mengoptimalkan suatu proses produksi dari awal mendapatkan bahan baku sampai pada pengangkutan (deliveri) produk jadi (akhir)nya. Salah satu cara atau metode ataupun Algoritma yang dapat mendukung proses tersebut adalah Minimum Spanning Tree (MST) yaitu jaringan yang menghubungkan sejumlah titik sehingga semuanya terkoneksi baik langsung ataupum tidak langsung dengan total lintasan minimum. Dalam penelitian ini, akan dikembangkan suatu Algoritma yang dapat memberikan informasi bahwa problem atau persoalan (MST) yang dihadapi mempunyai solusi tidak tunggal tetapi tidak langsung memberikan solusi MSTnya untuk masing-masing alternative solusi. Algoritma tersebut adalah suatu kombinasi antara Algoritma yang didasari dari Sistem Aljabar Gondran dan Algoritma Prim. Untuk mendapatkan solusi alternative masing-masing solusi dari persoalan MST yang tidak tunggal maka Algoritma yang dikembangkan dengan pengkombinasian tersebut akan diperoleh seluruh solusi yang mungkin dari problem MST tersebut.

Kata Kunci: Minimum Spanning Tree, Algoritma Prim, Aljabar Gondran, Solusi tidak tunggal.

1. Pendahuluan

1.1. Latar Belakang

Problem Minimum Spanning Tree (MST) adalah masalah menentukan jaringan yang menghubungkan sejumlah titik sehingga semuanya terkoneksi baik lansung ataupun tidak

langsung dengan total lintasan minimum. Pada MST diperbolehkan langkah surut dan satu titik bisa terkoneksi ke lebih dari dua titik. Masalah ini dapat digambarkan seperti pada jaringan kabel listrik yang menghubungkan bola lampu secara serial dan paralel.

Ada beberapa algoritma yang dapat digunakan untuk mendapatkan MST, antara lain algoritma Prim, Kruskal, Boruvka dan lain-lain. Tetapi algoritma ini hanya memberikan solusi tunggal meskipun persoalan MST yang dihadapi mempunyai beberapa alternatif solusi dengan nilai MST yang sama.

Banyak keuntungan yang bisa didapat jika seluruh alternative solusi yang mungkin pada suatu persoalan MST bisa diperoleh. Misalnya bisa ditambahkan kriteria baru sehingga solusi yang duperoleh lebih dekat kepada kebutuhan dan realitas dunia nyata.

Djauhari mengembangkan metode ini dengan melihatnya dari struktur Aljabar Gondran. Aljabar akan membantu melihat masalah ini secara lebih terstruktur dan fenomena yang ada lebih bisa digambarkan secara sistematis. Djauhari memberikan cara mendapatkan alternatif MST tetapi harus menggunakan grafik untuk menentukan alternatif yang muncul dengan cara membuang lintasan yang tidak perlu. Metode yang menggunakan Aljabar Gondran ini akan menjadi tidak praktis kalau jumlah titik yang harus diproses mencapai ratusan ataupun jarak yang digunakan bukan merupakan jarak fisik, seperti jarak yang berbentuk waktu.

Prim memberikan metode mendapatkan MST tetapi tidak bisa digunakan untuk mendapatkan alternatif solusi. Penggunaan Prim tanpa Aljabar Gondran untuk mendapatkan alternatif bisa berakibat munculnya loop baru atau kumpulan titik tersebut menjadi tidak terkoneksi. Penggunaan Aljabar Gondran tanpa Prim berakibat tidak bisa diketahuinya MST dari persoalan tersebut.

Kebutuhan untuk mencari metode yang mudah digunakan dan bisa memberikan seluruh solusi alternatif jika ada, makin lama makin dibutuhkan terutama dalam menekan biaya produksi dan efesiensi pada suatu industri mengingat metode yang ada selama ini belum dapat memberikan seluruh alternatif solusi yang mungkin.

Ada banyak metode dan Algoritma yang telah dikembangkan untuk mendapatkan MST dengan berbagai metode heuristik maupun analitik. Askin (1993) menguraikan MST sebagai salah satu kasus dari Travelling Salesman Problem (TSP). Hilier dan Lieberman (1990) memberikan metode heeuristik untuk memperoleh MST. Metode ini hanya efektif untuk kasus berskala kecil, dan metode ini tidak memberikan alternatif solusi jika solusi tidak tunggal.

1.2. Perumusan Masalah

Dalam penelitian ini, dapat dijabarkan beberapa perumusan masalah yang ada: (1) Bagaimana membentuk kombinasi antara Algoritma Prim dan Algoritma Djauhari yang didasarkan pada sistem Aljabar Gondran untuk mendapatkan seluruh alternatif solusi yang mungkin yang memberikan MST yang total lintasannya sama? (2) Bagaimana performansi kombinasi ini diukur dari jumlah solusi alternative yang bisa dihasilkan?

1.3. Tujuan Penelitian

Tujuan penelitian ini adalah seperti berikut: (1) Mempelajari algoritma dan metode yang ada dalam mendapatkan MST. (2) Mempelajari metode mendapatkan MST dari sudut pandang Graph. (3) Mempelajari metode mendapatkan MST dari tinjauan aljabar Gondran. (4) Memperoleh metode yang bisa digunakan umtuk mendapatkan solusi alternatif MST dari persoalan yang mempunyai solusi tidak tunggal.

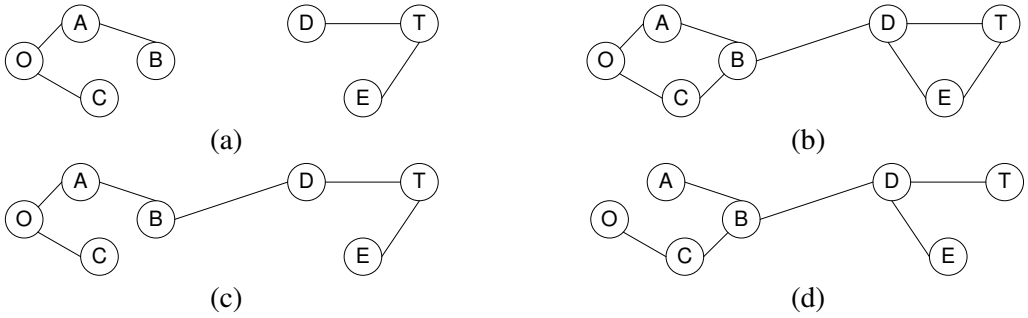
1.4. Manfaat Penelitian

Kegunaan kombinasi antara Algoritma Djauhari dan Algoritma Prim adalah sebagai suatu metode untuk memperoleh seluruh alternatif MST dengan mudah dan cepat. Dengan metode ini akan memperkaya studi tentang MST.

2. Tinjauan Pustaka

2.1. *Minimum Spanning Tree*

Spanning Tree adalah kumpulan *N-1* lintasana yang menghbungkan *N* titik. Dalam spanning tree ini tidak boleh terjadi siklus tetapi langkah surut diperbolehkan. Suatu S*panning Tree* ini disebut *Minimum Spanning Tree* (MST) jika jumlah jarak dari lintasannya adalah minimum.



Gambar 1. Ilustrasi konsep S*panning Tree*

Gambar 1 menggambarkan konsep *spanning tree*. Gambar 1. a bukan merupakan *spanning tree* karena titik-titik (O, A, B, C) tdak terkoneksi dengan titik-titik (D, E, T). Jaringan pada gambar 1. a ini biasa dianggap sebagai dua buah *spanning tree*. Pada gambar 1. b seluruh titik terkoneksi tapi bukan merupakan 'tree' karena adanya siklus (O-A-B-C-O) dan (D-T-E-D). Sedangkan gambar 1.c dan 1. d merupakan spanning tree. Ada banyak *spanning tree* yang mungkin dibuat. Untuk *n* titik maka akan ada n^{n-2} *spanning tree* yang berbeda yang bisa dibuat. *Minimum Spanning Tree* (MST) adalah *spanning tree* yang mempunyai total lintasan terkecil. Jika ada beberapa *spanning tree* yang memberikan nilai MST yang sama maka persoalan MST tersebut mempunyai solusi tidak tunggal.

2.2. *Minimum Spanning Tree* Sebagai Graph Berbobot Terkoneksi

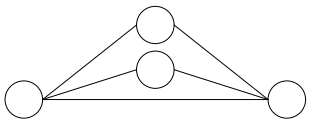
Sebuah graph G adalah himpunan tak kosong *V(G)* yang berisi objek yang disebut dengan vertices (titik/node) dan himpunan *E(G)* yang merupakan subset berelemen dua dari V(G) yang disebut edge (garis/lintasan).

Setiap graph mempunyai diagram. Diagram ini berguna untuk memahami permasalahan graph tersebut. Kadang-kadang diagram dari sebuah graph dipakai untuk menyatakan graph itu sendiri. Jumlah vertices dari sebuah graph G disebut orde dan jumlah edgenya disebut ukuran graph tersebut. Sebuag graph dengan orde *p* dan ukuran *q* disebut (*p, q*) graph. Walk (lintasan) dalam sebuah graph adalah urutan: $W : v_0, e_1, v_1, e_2, v_2, \cdots, v_{n-1}, e_n, v_n \quad (n \geq 0)$.

dari titik dan garis dimulai, dimulai dan diakhiri dengan titik sedemikian sehingga $ei = v_{i-1} - v_i$ untuk $i = 1, 2, \cdots, n$. Cara menyatakan walk *W* di atas juga bias dinyatakan secara lebih sederhana dengan $W : v_0, v_1, v_2, \cdots, v_{n-1}, v_n \quad (n \geq 0)$

Karena dua titik bisa menyatakan garis yang menghubungkannya.

Karena *W* dimulai dengan v_0 dan diakhiri dengan v_n maka W dinyatakan sebagai Walk $v_0 - v_n$. W juga dinyatakan mempunyai panjang n karena W menjalani n garis yang tidak perlu berbeda. Ada beberapa walk dengan sifat tertentu. Antara lain Trail yang merupakan walk tanpa pengulangan garis yang dilewati dan Path yang merupakan walk tanpa pengulangan titik yang dilewati. Dengan demikian maka sebuah path pasti merupakan trail dan tidak sebaliknya.



Gambar 2. Contoh Trail yang bukan Path

Siklus adalah walk $v_0, v_1, v_2, \cdots, v_{n-1}, v_n$ dengan $n \geq 3$ dan $v_0 = v_n$ dan *n* titiknya berbeda. Jika u dan v adalah titik pada grap *G* maka dikatakan u terkoneksi ke *v* jika graph *G* mengandung path *u – v*. Graph *G* dinyatakan sebagai graph terkoneksi jika *u* terkoneksi ke *v* untuk setiap pasangan *u* dan *v* titik-titik dari *G*.
Tree terdefinisikan sebagai graph terkoneksi tanpa siklus. Sebuah graph (terkoneksi atau tidak) tanpa siklus disebut forest. Sehingga setiap komponen dari fores adalah mrupakan tree. Spanning tree dari sebuah graph G adalah spanning subgraph dari G yang merupakan tree.

Teorema 1
Sebuah tree dengan orde *p* mempunyai ukuran *p-1*

Teorema 1 bisa dibuktikan dengan induksi. Bukti selengkapnya dari teorema ini bias dilihat pada Chartrand (1993).

Teorema Akibat 2
Jika G merupakan graph dengan orde p maka pernyataan berikut ini ekivalen:
(i) G adalah tree
(ii) G graph terkoneksi dengan ukuran *p-1*
(iii) G mempunyai ukuran p-1 dan tidak mengandung siklus.
Bukti selengkapnya dari teorema ini bias dilihat pada Chartrand (1993).

Sebuah graph G yang setiap garisnya diberikan bobot maka graph tersebut disebut graph berbobot G. Problem *Minimum Spanning Tree* (MST) adalah persoalan menentukan spaning tree dengan total bobot minimum pada suatu graph berbobot terkoneksi.

Teorema 3 (Formula Tree dari Cayley)
Jika G adalah graph yang diberi label dengan orde p, maka ada p^{p-2} spanning tree dari G yang berbeda.
Bukti selengkapnya dari teorema ini bias dilihat pada Chartrand (1993).

2.3. Sistem Aljabar Gondran

Pengertian tentang operasi biner dan system aljabar.

Definisi 1
Jika S adalah suatu himpunan yang tak kosong. Suatu pemetaan

$\varphi : S \; x \; S \quad \text{- - - - - - - -} \blacktriangleright \quad S$

$(x, y) \quad \text{- - - - - - - -} \blacktriangleright \quad x \nabla y$

dinamakan operasi biner ∇ pada S.
Jadi operasi biner adalah suatu aturan yang mengaitkan setiap pasangan terurut (*x,y*) di (*S x S*) dengan suatu unsur $z = x \nabla y$ di *S*

Definisi 2
Suatu himpunan yang dlengkapi dengan dua operasi biner disebut dengan system aljabar.

Definisi 3
Jika N adalah himpunan bilangan-bilangan real non negatif yang yang dilengkapi dengan bilangan $e=+\infty$, ditulis $N = R^{+} \cup \{+\infty$, perasi $\oplus$ dan $*$ pada N didefinisikan sebagai berikut :

$x \oplus y = \min(x, y)$ $\forall x, y$ di N

$x * y = maks(x, y)$ $\forall x, y$ di N

maka $(N, \oplus, *)$ disebut aljabar Gondran.

Sifat 1
Jika $(N, \oplus, *)$ adalah aljabar Gondran dan x, y, z di N, maka berlaku sifat berikut:

1. $x \oplus (y \oplus z) = (x \oplus y) \oplus z$ dan $x * (y * z) = (x * y) * z$ (asosiatif)
2. $x \oplus y = y \oplus x$ dan $x * y = y * x$ (komutatif)
3. Ada elemen $e = +\infty$ dan *0* di *N* yang berlainan sehingga berlaku

$$x \oplus e = e \oplus x = x$$
$$x * 0 = 0 * x = x$$

e disebut elemen kesatuan terhadap operasi $\oplus$ dan 0 disebut elemen kesatuan terhadap operasi $*$.
Bukti sifat ini bisa dilihat pada Chartrand (1993).

Sifat 2
$x \oplus x = x$ dan $x * x = x$ untuk setiap *x* di *N*
Bukti sifat ini bias dilihat pada Chartrand (1993).

Operasi-operasi yang berlaku di N dapat diperluas dan berlaku pula di G, dimana elemen-elemen di G adalah matriks yang data ditulis sebagai:

$$X_{(n\,x\,n)} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ x_{n1} & x_{n2} & \cdots & x_{nn} \end{bmatrix}$$ atau disingkat $X = (x_{ij})_{n\,x\,n}$ dimana x_{ij} di G dan i, j = 1, 2, …,n.

Utuk selanjutnya G akan didefinisikan sebagai himpunan matriks dengan elemen sebagai yang didefinisikan di atas.

Definisi 4

Jika $A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$ dan $B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & ba_{22} & \cdots & b_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{bmatrix}$ dua buah matriks berukuran *(nxn)* di G. Perkalian dua buah matriks di G didefinisikan oleh:

$$AB = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & ba_{22} & \cdots & b_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{bmatrix} =$$

$$\begin{bmatrix} a_{11} * b_{11} \oplus a_{12}b_{21} \oplus \cdots \oplus a_{1n}b_{n1} & a_{11} * b_{12} \oplus a_{12}b_{22} \oplus \cdots \oplus a_{1n}b_{n2} & \cdots & a_{11} * b_{1n} \oplus a_{12}b_{2n} \oplus \cdots \oplus a_{1n}b_{nn} \\ a_{21} * b_{11} \oplus a_{22}b_{21} \oplus \cdots \oplus a_{2n}b_{n1} & a_{21} * b_{21} \oplus a_{22}b_{22} \oplus \cdots \oplus a_{2n}b_{n2} & \cdots & a_{21} * b_{1n} \oplus a_{22}b_{2n} \oplus \cdots \oplus a_{2n}b_{nn} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ a_{n1} * b_{11} \oplus a_{n2}b_{21} \oplus \cdots \oplus a_{nn}b_{n1} & a_{n1} * b_{12} \oplus a_{n2}b_{22} \oplus \cdots \oplus a_{nn}b_{n2} & \cdots & a_{n1} * b_{1n} \oplus a_{n2}b_{2n} \oplus \cdots \oplus a_{nn}b_{nn} \end{bmatrix}$$

Contoh 1

Jika $A = \begin{bmatrix} 2 & 5 \\ 4 & 1 \end{bmatrix}$ dan $B = \begin{bmatrix} 1 & 7 \\ 3 & 4 \end{bmatrix}$ dua matriks di G maka

$$AB = \begin{bmatrix} 2 & 5 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} 1 & 7 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 2*1 \oplus 5*3 & 2*7 \oplus 5*4 \\ 4*1 \oplus 1*3 & 4*7 \oplus 1*4 \end{bmatrix} = \begin{bmatrix} 2 \oplus 5 & 7 \oplus 5 \\ 4 \oplus 3 & 7 \oplus 4 \end{bmatrix} = \begin{bmatrix} 2 & 5 \\ 3 & 4 \end{bmatrix}$$

Definisi 5

Jika $A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$ dan $B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & ba_{22} & \cdots & b_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{bmatrix}$ dua buah matriks berukuran (nxn) di G. Penjumlahan dua buah matriks di G didefinisikan oleh :

$$AB = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & ba_{22} & \cdots & b_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{bmatrix} =$$

$$\begin{bmatrix} a_{11} \oplus b_{11} & a_{12} \oplus b_{12} & \cdots & a_{1n} \oplus b_{1n} \\ a_{21} \oplus b_{21} & a_{22} \oplus b_{22} & \cdots & a_{2n} \oplus b_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ a_{n1} \oplus b_{n1} & a_{n2} \oplus b_{n2} & \cdots & a_{nn} \oplus b_{nn} \end{bmatrix}$$

Contoh 2

Jika $A = \begin{bmatrix} 2 & 5 \\ 4 & 1 \end{bmatrix}$ dan $B = \begin{bmatrix} 1 & 7 \\ 3 & 4 \end{bmatrix}$ dua matriks di G $AB = \begin{bmatrix} 2 & 5 \\ 4 & 1 \end{bmatrix}\begin{bmatrix} 1 & 7 \\ 3 & 4 \end{bmatrix}$

maka

$$A + B = \begin{bmatrix} 2 & 5 \\ 4 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 7 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 2 \oplus 1 & 5 \oplus 7 \\ 4 \oplus 3 & 1 \oplus 4 \end{bmatrix} = \begin{bmatrix} 1 & 5 \\ 3 & 1 \end{bmatrix}$$

Sifat 3

$A + B = B + A$ untuk setiap matriks *A, B* di *G*

Perkalian matriks di G tidak komutatif . $AB \neq BA$

Sifat 4

$A \oplus (B \oplus C) = (A \oplus B) \oplus C$ untuk setiap matriks *A, B, C* di *G*.

Bukti sifat ini pada Sudrajat (1989).

Sifat 5

$A * (B * C) = (A * B) * C$ untuk setiap matriks *A, B, C* di *G*.

Bukti sifat ini pada Sudrajat (1989).

Sifat 6

$A * (B \oplus C) = (A * B) \oplus (A * C)$ untuk setiap matriks *A, B, C* di *G*.

Bukti sifat ini pada Sudrajat (1989).

Definisi 6

Jika S suatu himpunan yang tak kosong dengan operasi biner ∇, maka himpunan S dengan operasi biner ∇ ditulis (S, ∇) yang bersifat asosiatif disebut semi-group.

Semi-group (S, ∇) disebut komutatif jika $x \nabla y = y \nabla x$ untuk setiap *x, y* di *S*

Sifat 7

$(G, \oplus)$ adalah semi-group komutatif.

Bukti sifat ini pada Sudrajat (1989).

Definisi 7

Semi group (S, ∇) dengan unsur kesatuan *u* di S disebut monoid dan ditulis dengan (S, ∇, u).

Sifat 8

$(G, \oplus, e)$ adalah monoid,

Bukti sifat ini pada Sudrajat (1989).

Definisi 8
Jika R suatu himpunan tak kosong dengan operasi + dan . , maka sistem aljabar $(R,+,.)$ disebut semi-ring jika dipenuhi:

1. $(R,+)$ semi-group komutatif
2. $(R,.)$ semi-group
3. $x\,.\,(y+z) = x\,.\,y + x\,.\,y$ untuk semua x, y, z di R.

Semi-ring $(R,+,.)$ disebut komutatif jika $x\,.\,y = y\,.\,x$ untk semua x, y di R.

Sifat 9
$(G, \oplus, *)$ adalah semi-ring komutatif dengan unsure kesatuan.
Bukti sifat ini pada Sudrajat (1989).

3. Metodologi Penelitian

3.1. Studi Literatur

Studi literatur dilakukan untuk mempelajari masing-masing algoritma dari Prim dan Djauhari. Pendekatan yang dulakukan oleh Prim dan Djauhari adalah dari sisi yang berbeda. Prim melakukan pendekatan melalui Graph sedangkan Djauhari melakukan pendekatan dengan Aljabar khususnya system Aljabar Gondran. Kedua algoritma tersebut kemudian diimplementasikan ke dalam program computer untuk mempercepat dan mempermudah proses komputasi.

3.2. Algoritma Untuk mendapatkan MST

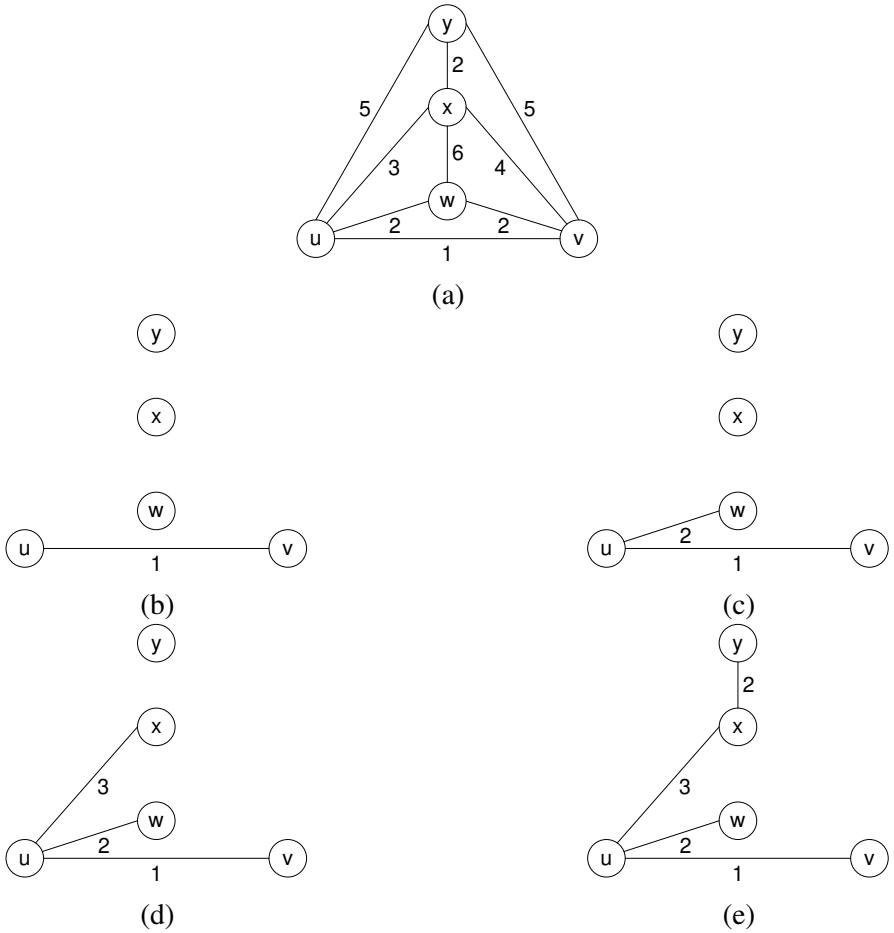
Pada bagian ini diberikan dua algoritma yang akan menjadi bahan pembahasan.

3.2.1. Algoritma Prim

Algoritma Prim menyelesaikan persoalan MST dengan memandangnya sebagai graph berbobot terkoneksi. Algoritma ini hanya berlaku utuk graph berbobot terkoneksi nontrivial, yaitu yang jumlah nodenya (titiknya) lebih besar dari satu. Tahapan penyelesaian Masalah MST untuk graph berbobot terkoneksi (p, q) G.

Langkah 1: (Inisialisasi tree T). Ambil u sembarang titik dari G dan $T \leftarrow u$. Langkah 2: (Tree T di update). Misalkan e adalah garis dengan bobot minimum yang menghubungkan sebuah titik di T dan sebuah titik yang tidak di T, dan $T \leftarrow T + e$. Langkah 3: (Untuk memeriksa terbentuknya MST). Jika ukuran dari $T = p - 1$ maka MSTnya adalah $E(T)$. Jika tidak kembali ke langkah kedua.

Contoh 3.
Jika Algoritma Prim diterapkan pada graph di gambar 3 (a) maka tahapan pembentukan MST sesuai dengan urutan pada Algoritma Prim bisa dilihat pada gambar 3 (b) sampai dengan gambar 3 (e).
Pada gambar 3 (b) T diinisialisasi dengan mengambil titik u. Pada langkah 2, dicari garis dengan bobot minimum yang menghubungkan sebuah titik di luar T dan sebuah titik di dalam T. Yang memenuhi syarat adalah garis uv dengan bobot 1. Kemudian titik v dimasukkan ke dalam T dengan membentuk tree uv. Sekarang T mempunyai 2 titik dan 1 garis. Pada langkah 3 dperiksa apakah jumlah garis sama dengan 4. Karena belum maka langkah 2 kembali dilakukan dan seterusnya seperti pada gambar 3 (c) sampai dengan gambar 3 (e). Implementasi dari algoritma Prim akan berjalan baik sekali apabila graph yang dimiliki merupakan grap lengkap.



Gambar 3. Langkah Pada Algoritma Prim

3.2.2. Algoritma Djauhari

Algoritma Djauhari didasarkan pada system Aljabar Gondran. Pada system Aljabar Gondran ada perkalian matriks seperti pada perkalian matriks umum, tetapi dengan mengganti operasi perkalian dua elemen adalah maksimum dari dua elemen tersebut, dan operasi penjumlahan dua elemen hasilnya adalah minimum dari kedua elemen tersebut.

Langkah 1: Bentuk matriks disimilaritas atau matriks jarak $D_{p\,x\,p}$ dari p titik. Semakin dekat hubungan antar elemen maka jaraknya semakin kecil. Matriks disimilaritas bisa diperoleh dari negasi matriks similaritas. Inisialisasi $D^0 = D$. Matriks ini adalah merupakan matriks simetris. Langkah 2: Hitung $D^2 = D * D$. Langkah 3: Jika $D^2 = D$ lanjutkan langkah 4. Jika tidak maka $D = D^2$ dan kembali ke langkah 2. Langkah 4: Hitung $D - D^0$ dan buang diagonal utama dan salah satu segitiga atas atau bawah, karena matriks ini simetris dan elemen diagonal utamanya nol. Jika elemen nol matriks $D - D^0$ sama dengan p -1 maka persoalan mempunyai MST yang unik. Jika tidak maka ada alternative solusi. Elemen nol merupakan garis yang menghubungkan kedua titik yang berkaitan pada matriks jarak/disimilaritas.

Iterasi pada algoritma di atas sebenarnya adalah iterasi mencari $D, D^2, D^4, D^8, D^{16}, \cdots$ Sebenarnya iterasi yang dilakukan bisa dengan mencari $D, D^2, D^3, D^4, D^5, \cdots$ Tetapi cara ini tidak praktis dan lama. Karena untuk mendapatkan D^{16} bisa dengan empat langkah dan tidak perlu dengan 16 langkah. Algoritma ini adalah algoritma optimal. Bukti keoptimalan dari algoritma ini ada pada Dadang (1989).

3. 3. Algoritma Untuk Mendapatkan MST

Kombinasi Algoritma Prim dan Djauhari dimaksudkan untuk mempemudah proses mendapatkan seluruh solusi MST yang mungkin dari suatu persoalan MST yang mempunyai solusi yang tidak tunggal. Algoritma Djauhari memberikan indikator bahwa persoalan MST yang dihadapi mempunyai solusi yang tidak tunggal jika persoalan MST yang dihadapi memang mempunyai solusi tidak tunggal. Tetapi Algoritma Djauhari tidak memberikan nilai MST dari persoalan tersebut. Algoritma Prim menghasilkan nilai MST tetapi hanya memberikan satu solusi meskipun persoalan MST yang dihadapi mempunyai solusi tidak tunggal. Pada kombinasi kedua algoritma ini output dari algoritma Prim dijadikan patokan dalam melakukan kombinasi garis hasildari algoritma Djauhari. Untuk itu table garis dari algoritma Prim dan Djauhari harus sort.

Selengkapnya kominasi kedua algoritma adalah sebagai berikut: Tahap 1 (Algoritma Prim). Langkah 1: (Inisialisasi tree T). Ambil *u* sembarang titik dari *G* dan $T \leftarrow u$. Langkah 2: (Tree T di update). Misalkan *e* adalah garis dengan bobot minimum yang menghubungkan sebuah titik di *T* dan sebuah titik yang tidak di T dan $T \leftarrow T + e$. Langkah 3: (Memeriksa terbentuknya MST). Jika ukuran dari T = p-1 maka MSTnya adalah E(T). Jika tdak kembali ke langkah kedua. Langkah 4: (Pengurutan baris kolom). Urutkan table garis baris dan kolom sedemikian sehingga indeks baris lebih besar dari indeks kolom untuk setiap garis. Langkah 5: (Pengurutan garis). Urutkan table garis baris dan kolom hasil pada langkah 4 sehingga terurut menurut kolombaris dari kecil ke besar. Tahap 2 (Djauhari). Langkah 1. Gunakan matriks jarak *D* pada tahap 1 di atas. Inisialisasi $D^0 = D$. Langkah 2. Hitung $D^2 = D * D$. Langkah 3. Jika $D^2 = D$, lanjutkan langkah 4. Jika tidak, maka $D = D^2$ dan kembali ke langkah 2. Langkah 4. Hitung elemen nol pada matriks $D - D^0$ yang telah dibuang elemen diagonal utama dan salah satu segitiga atas atau bawah. Jika solusi unik selesai, jika tidak lanjutkan langkah 5. Langkah 5. Hilangkan entri pada table garis hasil tahap 2 (Djauhari) yang sama dengan entri pada hasil tahap 1 (Prim) dan sebut table selisih. Langkah 6. Untuk setiap garis pada table selisih, cari entri pada table garis Prim yang berpadanan baik pada baris atau kolom. Setiap entri yang berpadanan bisa dipertukarkan untuk memperoleh alternative solusi.

4. Pengujian dan Pembahasan

4.1. Proses Pembuatan Data Simulasi

Kombinasi Algoritma Prim dan Djauhari dimaksudkan untuk memperoleh seluruh alternatif solusi dari suatu persoalan MST yang mempunyai solusi tidak tunggal. Karena sulitnya mendapatkan data yang mempunyai solusi tidak tunggal maka untuk implementasi digunakan data simulasi. Data simulasi yang digunakan dalam implementasi ini dihasilkan secara random dengan bantuan komputer. Pertama-tama dibangkitkan titik-titik random dalam bentuk koordinat, yang absis dan ordinatnya dihasilkan secara random. Kemudian dari titik-titik ini diukur jarak Euclid antar titik sehingga diperoleh matriks jarak

4.2. Proses Perhitungan dan Komputasi

Untuk melakukan proses perhitungan, implementasi algoritma tersebut dibuat menggunakan program MATLAB.06. Karena keterbatasan jumlah variable yang bisa didefinisikan dalam array maka implementasi diterapkan sampai data berukuran maksimum 120 titik/vertice. Untuk *p* titik maka akan diperlukan p^2 variable.

4.3. Implementasi Metode

Pada bagian ini akan dijelaskan implementasi metode pada 3 (tiga) kasus/permasalahan

dengan data simulasi. Adapun permasalahan pertama digunakan sebanyak 40 titik, permasalahan kedua digunakan 75 titik dan permasalahan ketiga digunakan 110 titik.

4.3.1. Kasus 40 vertice (titik)

Dengan menggunakan algoritma Prim diperoleh tabel edge/garis seperti tabel 1, dimana Algoritma Prim mmberikan solusi total MST sebesar 145.5. Setelah dilakukan sorting kolom dan kolom maka diperoleh tabel 2.

Tabel 1. Hasil algoritma Prim pada Kasus 40 vertice

No.	BARIS	KOLOM	No.	BARIS	KOLOM
1	1	5	21	21	22
2	1	7	22	23	28
3	2	3	23	24	31
4	4	2	24	25	20
5	6	4	25	25	29
6	6	9	26	26	25
7	7	10	27	27	21
8	8	6	28	28	32
9	10	8	29	29	36
10	10	11	30	30	27
11	11	13	31	31	26
12	13	16	32	31	37
13	14	12	33	32	33
14	14	18	34	32	34
15	15	14	35	33	30
16	16	17	36	33	40
17	17	15	37	36	39
18	17	23	38	37	38
19	18	24	39	39	35
20	20	19			

Tabel 2. Hasil Pengurutan tabel 1

No.	BARIS	KOLOM	No.	BARIS	KOLOM
3	3	2	19	24	18
4	4	2	24	25	20
1	5	1	26	26	25
5	6	4	27	27	21
2	7	1	22	28	23
8	8	6	25	29	25
6	9	6	30	30	27
7	10	7	23	31	24
9	10	8	31	31	26
10	11	10	28	32	28
11	13	11	33	33	32
12	13	16	35	33	30

13	14	12	34	34	32
15	15	14	29	36	29
16	17	16	32	37	31
17	17	15	38	38	37
14	18	14	37	39	36
20	20	19	39	39	35
21	22	21	36	40	33
18	23	17			

Hasil dari Algoitma Djauhari adalah tabel edge/garis seperti tabel 3. Total bobot dari Algoritma Djauhari adalah 153.3. Selisih dari table Djauhari dan Prim adalah tabel 4. Dapat diperiksa pada table Djauhari dan dicari entri yang berpadanan dengan baris 27 atau kolom 22 dengan bobot 7.8 maka diperoleh garis nomor 24.

Tabel 3. Hasil dari Algoitma Djauhari pada Kasus 40 vertice

No.	BARIS	KOLOM	BOBOT	No.	BARIS	KOLOM	BOBOT
1	3	2	2.7	21	24	18	6.3
2	4	2	4	22	25	20	5.7
3	5	1	4.6	23	26	25	3.1
4	6	4	4.9	24	27	21	7.8
5	7	1	7.5	25	27	22	7.8
6	8	6	3.7	26	28	23	6.6
7	9	6	3.5	27	29	25	2.8
8	10	7	5.4	28	30	27	2.6
9	10	8	3.9	29	31	24	6.5
10	11	10	0.9	30	31	26	4.8
11	13	11	4.3	31	32	28	2
12	14	12	0.7	32	33	30	2.4
13	14	12	1.1	33	33	32	4.2
14	15	14	2.7	34	34	32	3
15	17	16	4.5	35	36	29	4.5
16	17	15	1.7	36	37	31	4.1
17	18	14	5.6	37	38	37	6
18	20	19	0.4	38	39	35	1.6
19	22	21	0.3	39	39	36	0.9
20	23	17	3.7	40	40	33	4.5

Tabel 4. Selisih dari tabel Djauhari dan Prim pada Kasus 40 vertice

No.	BARIS	KOLOM	BOBOT
25	27	22	7.8

4.3.2. Kasus 75 vertice (titik)

Dengan menggunakan Algoritma Prim diperoleh tabel 5, total lintasan 193.7. Dengan menggunakan Algoritma Djauhari diperoleh tabel 6. Total lintasan minimumnya adalah 202.3. Tabel selisihnya dapat dilihat pada tabel 7. Bisa dilihat dari table Djauhari bahwa garis 12 bisa menjadi alternative untuk garis 13 karena kolomnya berpadanan dan bobotnya sama. Sedangkan gars 64 menjadi alternative untuk garis 63 karena barisnya berpadanan dan bobotnya sama.

Tabel 5. Hasil algoritma Prim pada Kasus 75 vertice

No.	BARIS	KOLOM	No.	BARIS	KOLOM
4	5	4	29	40	30
1	7	1	43	41	36
5	7	5	41	43	39
6	8	2	38	44	37
7	9	3	44	44	42
3	10	4	42	45	40
8	10	9	45	46	44
2	11	1	39	47	38
10	12	8	48	48	41
11	13	12	47	49	47
9	14	10	51	49	48
13	16	6	49	50	48
14	17	16	50	51	48
12	18	14	46	52	46
15	19	17	52	54	50
17	19	15	53	55	52
18	20	13	56	55	53
16	21	18	54	56	54
21	22	17	57	57	55
20	25	21	59	58	56
22	26	25	55	59	54
23	27	26	61	60	58
25	27	22	62	61	58
26	28	20	60	62	57
27	29	24	58	63	55
28	29	28	64	65	61
19	30	20	65	66	62
30	31	23	68	66	64
31	32	31	69	67	66
24	33	26	66	68	63
35	35	32	63	69	59
36	36	34	67	70	65
37	36	35	70	71	67
32	37	33	73	72	71
33	38	33	72	73	70
34	39	34	71	74	69
40	39	29	74	75	73

Tabel 6. Hasil algoritma Djauhari pada Kasus 75 vertice

No.	BARIS	KOLOM	BOBOT	No.	BARIS	KOLOM	BOBOT
1	5	4	1.6	39	40	30	4.3
2	7	1	3.1	40	41	36	2.1
3	7	5	1.4	41	43	39	2.4
4	8	2	4.6	42	44	37	3.3
5	9	3	2.7	43	44	42	1.8
6	10	4	2.3	44	45	40	2.3
7	10	9	5	45	46	44	1.3
8	11	1	3.8	46	47	38	4.4

9	12	8	2	47	48	41	2.7
10	13	12	1.6	48	49	47	1.5
11	14	10	1.6	49	49	48	3.1
12	15	6	4.2	50	50	48	2.8
13	16	6	4.2	51	51	48	3
14	17	16	1.7	52	52	46	4.1
15	18	14	2.5	53	54	50	5.5
16	19	15	2.3	54	55	52	2.8
17	19	17	1.7	55	55	53	3.2
18	20	13	2.5	56	56	54	3.3
19	21	18	0.9	57	57	55	0.5
20	22	17	2.7	58	58	56	0.4
21	25	21	1.9	59	59	54	2.7
22	26	25	3	60	60	58	1.8
23	27	22	2.4	61	61	58	3.7
24	27	26	2.1	62	62	57	3.9
25	28	20	3.6	63	63	55	4.4
26	29	24	2.1	64	63	57	4.4
27	29	28	2.2	65	65	61	2.3
28	30	20	3.8	66	66	62	4.1
29	31	23	3	67	66	64	0.7
30	32	31	1.3	68	67	66	1
31	33	26	4.1	69	68	63	2.5
32	35	32	2.2	70	69	59	4.2
33	36	34	1.2	71	70	65	2
34	36	35	0.2	72	71	67	1.7
35	37	33	1.5	73	72	71	1.1
36	38	33	1.6	74	73	70	1.5
37	39	29	4	75	74	69	2.4
38	39	34	3.2	76	75	73	5.3

Tabel 7. Selisih dari tabel Djauhari dan Prim pada Kasus 75 vertice

No.	BARIS	KOLOM	BOBOT
12	15	6	4.2
64	63	57	4.4

4.3.3. Kasus 110 vertice (titik)

Untuk kasus ini diperoleh hasil dari algoritma Prim dengan total lintasan 237.9. Sedangkan hasil dari Algoritma Djauhari memberikan nilai 257.6 dengan tabel 8. Selisihnya ada pada tabel 9.

Tabel 8. Hasil algoritma Djauhari pada Kasus 110 vertice

No.	BARIS	KOLOM	BOBOT	No.	BARIS	KOLOM	BOBOT
1	7	1	1.5	59	59	58	1.3
2	7	3	2	60	60	54	2.6
3	8	2	1.7	61	60	56	3.2
4	11	4	1.8	62	61	56	3.2
5	11	10	1.3	63	61	60	0.8
6	12	5	4.3	64	62	52	4

7	13	3	3.2	65	62	53	4
8	13	8	2.6	66	64	63	0.3
9	14	11	1.8	67	65	62	1.1
10	15	6	2.3	68	66	62	2.5
11	15	8	2.9	69	67	55	4.3
12	16	5	4	70	67	63	3.9
13	17	7	2.6	71	68	65	1.9
14	17	16	2.1	72	69	66	1.6
15	18	9	2	73	70	61	3.4
16	18	10	2.3	74	71	69	1.4
17	20	15	3.4	75	72	68	2.9
18	22	18	2.6	76	73	67	2.5
19	22	20	3.4	77	75	71	1.9
20	24	17	3.5	78	75	74	1.5
21	24	21	3.3	79	76	69	2.5
22	27	26	0.8	80	77	70	2.4
23	28	22	1.8	81	79	77	1
24	28	23	1.1	82	80	73	2.8
25	29	28	0.6	83	81	74	1.6
26	30	25	1.3	84	81	78	2.9
27	30	27	1.6	85	82	76	1.8
28	31	19	2.9	86	84	79	3
29	32	27	1.6	87	85	81	1.6
30	32	30	0.7	88	86	85	0.4
31	34	20	3.3	89	87	83	1.5
32	34	33	1.6	90	88	82	1
33	35	31	1.2	91	89	80	2.9
34	35	33	1.2	92	90	89	0.6
35	36	34	3.2	93	91	80	2.9
36	37	32	2.6	94	91	89	0.6
37	38	32	2.6	95	92	90	0.9
38	38	37	0	96	93	88	1.8
39	39	36	0.5	97	94	85	1.8
40	40	38	1.8	98	95	83	3.1
41	41	39	1.6	99	95	93	3.1
42	43	42	3.5	100	96	91	2.2
43	44	40	2	101	97	92	2.2
44	47	41	1.5	102	97	94	2.3
45	47	46	4	103	98	84	4.6
46	48	41	2.8	104	99	87	3.1
47	49	43	2.3	105	100	98	0.6
48	49	45	1.4	106	101	96	2.3
49	50	48	2.4	107	102	97	1.4
50	51	43	2.3	108	103	95	2.4
51	52	49	1.8	109	103	99	3
52	53	51	2.2	110	104	93	3.7
53	53	52	0.6	111	106	99	1.9
54	54	44	3.7	112	106	105	1.2
55	55	46	3	113	107	103	1.2
56	57	51	3.1	114	109	108	3.7

57	58	54	2.3	115	110	104	2.3
58	59	57	3.9	116	110	109	1.6

Tabel 9. Selisih dari tabel Djauhari dan Prim pada Kasus 110 vertice

No.	BARIS	KOLOM	BOBOT
27	30	27	1.6
36	37	32	2.6
50	51	43	2.3
62	61	56	3.2
65	62	53	4
93	91	80	2.9
104	99	87	3.1

Alternatif yang mungkin adalah dengan mempertukarkan 27 dengan 29; 36 dengan 37; 50 dengan 47; 62 dengan 61; 65 dengan 64; 93 dengan 91. Sedangkan garis 104 dapat dihilangkan.
Dari 6 alternatif di atas maka akan ada $2^6 = 64$ alternatif tree yang mungkin dibuat.

5. Kesimpulan

(1) Algoritma Prim memberikan satu solusi Mimnimum Spanning Tree (MST) dan tidak memberikan solusi alternatif yang mungkin meskipun persoalan MST tersebut mempunyai solusi yang tidak tunggal. (2) Algoritma Djauhari memberikan cara sistematis untuk megetahui apakah persoalan MST tersebut mempunyai solusi alternatif. Meskipun memberikan solusi alternative, Algoritma Djauhari tidak memberikan secara pasti totallintasan minimum dari MST tersebut. (3) Kombinasi Algoritma dan Sistem Aljabar Gondran memberikan sekaligus MST dan alternatif solusi yang ada tanpa sekalipun harus memplot dalam bentuk gambar.

6. Saran

Kombinasi algoritma Djauhari dan Algoritma Prim kadang-kadang harus diplot untuk mendapatkan siklus yang terjadi dan kemudian menhilangkan *edge*/garis yang tidak perlu. Peluang dan pengembangan penelitian lebih lanjut adalah pengimplementasian Algoritma Kombinasi tersebut pada permasalahan teknologi kelompok yang benyak digunakan pada bidang manufaktur atau yang berkaitan dengan proses produksi.

Referensi

Ahmad, Shalahuddin., 1996, Pengembangan Algoritma Solusi Alternatif Minimum Spanning Tree dari Kombinasi Algoritma Prim dan Algoritma Gondran, Tesis, S2 Teknik Industri, ITB.

Askin, Ronald G., 1993, Modelling and Analysis of Manufacturing System, New York, John Willey, 185-190.

Chartrand, Gary and Oellermann, Ortrud R., 1993, *Applied and Algorithmic Graph Theory*, New York, McGraw-Hill, 63-96.

Chow W, W. dan Hawaleshka O., 1993, *Minimazing Intercellular Part Movements in Manufacturing Cell Formation*, Int. J. Production Research, Vol 31, No. 9., 2161-2170.

Djauhari, Maman A., 1991, *An Easy Way to Find All Possible MST*, Bandung, Laboratorium Statistik Jurusan Matematika, ITB.

Hillier, Frederick S dan Lieberman, Gerald J.m 1990, *Introduction to Operation Research*, New

York, McGraw-Hill, 341-346.

Hokey, Min and Dooyoung, Shin., 1993, *Simultaneus Formation of Machine and Human Cell in Group Technology : A Multiple Objective Approach*, Int. J. Production Research, Vol 31. No. 10, 2307-2318.

Luong, L.H.S., 1993, *A Cellular Similarity Coefficient Algorithm for The Design of Manufacturing Cells*, Int. J. Production Research, Vol 31, No. 8, 1757-1766..

Sudrajat, Dadang, 1989, *Klasifikasi Hirarki Dilihat dari Sruktur Aljabar Gondran*, Skripsi, Jurusan Matematika ITB.

Diktat Matematika Diskret., 1998, Program Studi Teknik Informatika, Fakultas Teknologi Industri, UAJY.